PyScanCf – The Open Source Python based Library for IMD Doppler Weather Radar Datasets

Syed H.A.[1,2], Sayyed I.[2], Kalapureddy M.C.R.[2], Grandhi K.K.[1]

1. Department of Atmospheric and Space Sciences, Savitribai Phule Pune University, Pune
2. PDTC, Indian Institute of Tropical Meteorology, Pune.

Corresponding Author email: hamidsyed37@gmail.com

Introduction

Weather radar observations include a large amount of information, but great effort is necessary to extract scientifically meaningful data. Weather radar data processing, rectification, and analysis span a wide range of computer professions and subjects. Given the breadth of information available from weather radars, it is vital to have software that is both adaptive and expandable[1]. Users of such software will have a diverse set of needs, ranging from simple data presentation to the creation of complex processing pipelines. Software should ideally be simple for normal tasks while simultaneously offering experienced users with access to more powerful capabilities.

Objectives

The PyScanCf library is discussed in this poster, which is an open-source Python package built on top of the scientific Python ecosystem modules that generates cf-radial and gridded data from single sweeps. As illustrated in Fig. 1, there is a dense network of IMD radars. IMD datasets typically include 10 files for a full volume scan inside its domain, i.e., ten files for ten sweeps. This package provides a Python framework for working with single sweep radar data, from which sophisticated workflows can be constructed and used as input to Python-based open-source software for processing radar data like as Py-ART [2], and Wradlib [3].

Architecture

The package is divided into three sub-modules based on the type of functionality. The first sub-module is called;

- "cfrad" and it builds Cf-radial [4] data in NetCDF format from numerous single sweeps. Cfrad's output data will be fed into Py-ART or Wradlib for further processing such as visualisation, clutter removal, and so on.
- The second submodule is "get grid," which employs Py-ART to convert polar coordinates to cartesian coordinates using the Barnes2 method [5].
- Following that, the third module, "plot cappi," is used to plot Max-Z and other Max-CAPPI Products. Max-CAPPI products include, for example, Max-Z, which indicates maximum reflectivity, Max-V, which represents maximum radial velocity, and Max-W, which represents maximum spectrum width at constant altitude plan position indicator (CAPPI).

Functionality

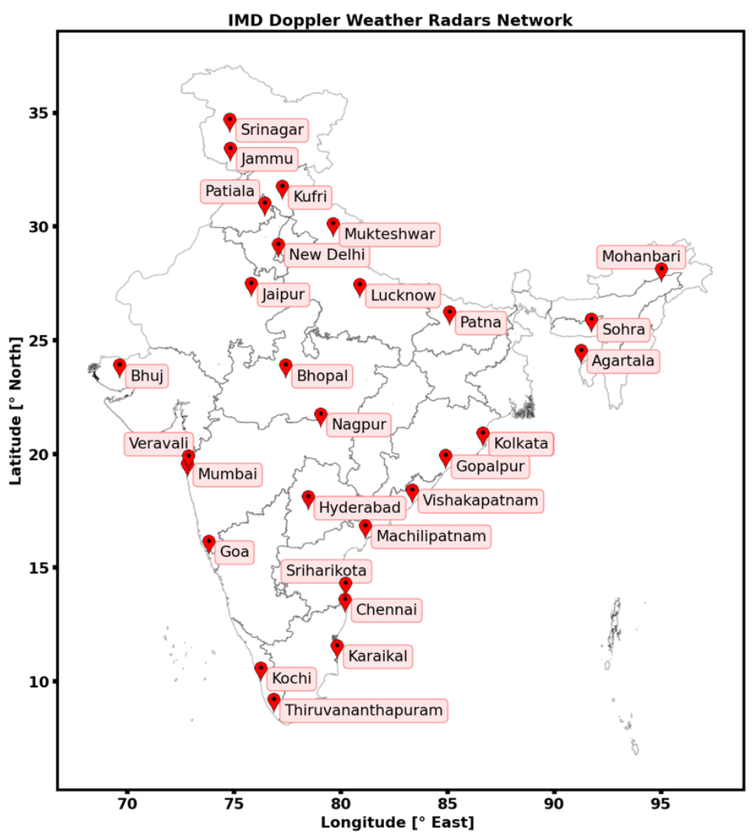

Fig. 1 - The network of thirty Doppler weather radars across Indian region.

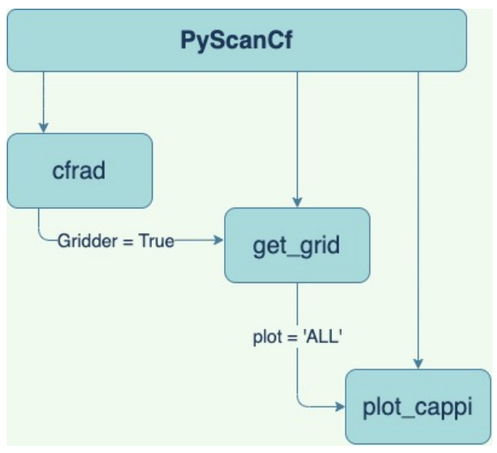

Fig. 2 – A simple flowchart of the PyScanCf toolkit.

Figure 2 depicts a simple flowchart of PyScanCf. If the attributes of the cfrad function are defined, the user can pass all the functions of submodules in a single command that is in cfrad, for example, cfrad("input dir", "output dir", gridder = True, plot= "ALL"). if gridder is True the module will call the get grid function to convert the data to a gridded format. Furthermore, if the plot function in the cfrad is not set to "None," the cfrad module will plot Max-CAPPI products such as Max-Z, Max-V, and Max-W because the default option is "ALL," but if the user only wants the plot of Max-Z, the user may simply input "REF".

Figures from weather radars are vital because they include a wealth of information regarding live weather conditions, which is especially important when it comes to extreme phenomena such as cyclones and tornadoes. Fig. 3 depicts Tropical Cyclone Tauktae using PyScanCf's plot_cappi function. The plane view of Fig. 3 depicts maximum reflectivity, commonly known as CAPPI display, whereas the top and right panels depict vertical profiles of maximum reflectivity at Cyclone Tauktae's axes one and two, respectively. The clouds virtually reach 20 km in height, the peak reflectivity is only seen below 10 km, and the eye is nearly 50 km in size.

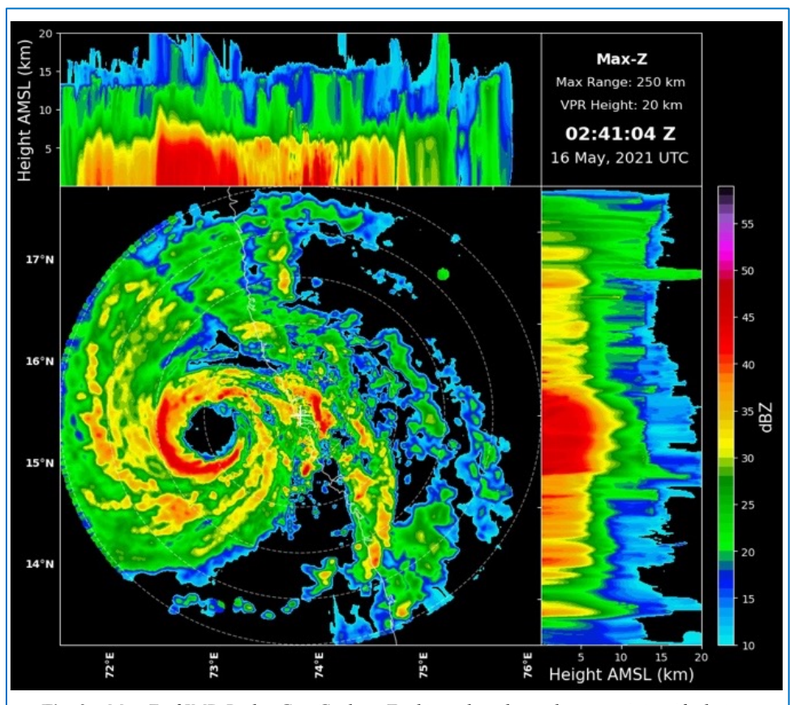

***Fig. 3** – Max-Z of IMD Radar Goa Cyclone Tauktae when the cyclone was intensified at 02:41 UTC, with its associated vertical profiles*

Availability

Operating system

- Linux, OS X, and Windows.

Programming language

- Python [3.7, 3.8 and 3.9]

Dependencies

- NumPy 1.10+,
- matplotlib 1.1+,
- NetCDF4 1.4+,
- Py-ART 1.6+,
- Pandas 1.3+,
- Cartopy 0.20 +

SOFTWARE LOCATION

Archive

Name: Zenodo

Persistent identifier:
https://doi.org/10.5281/zenodo.5574160

License: MIT

Publisher: Syed, Hamid Ali

Version published: 1.0.18

Date published: 19 November 2021

Code repository

Name: GitHub

Identifier:
https://github.com/syedhamidali/PyScanCf

License: MIT

Date published: 19 November 2021

References

1. Heistermann, M., S. Collis, M. J. Dixon, S. Giangrande, J. J. Helmus, B. Kelley, J. Koistinen, D. B. Michelson, M. Peura, T. Pfaff, and D. B. Wolff. " The Emergence of Open-Source Software for the Weather Radar Community". *Bulletin of the American Meteorological Society* 96.1 (2015): 117-128
2. Helmus, J.J. and Collis, S.M., 2016. The Python ARM Radar Toolkit (Py-ART), a Library for Working with Weather Radar Data in the Python Programming Language. Journal of Open Research Software, 4(1), p.e25
3. Heistermann, M., Jacobi, S. and Pfaff, T., 2013. An open source library for processing weather radar data (wradlib). Hydrology and Earth System Sciences, 17(2), pp.863-871.
4. Dixon, M., Lee, W., Rilling, B., Burghard, C. and Van Andel, J., 2015. CfRadial data file format: Proposed CF-compliant netCDF format for moments data for RADAR and LIDAR in radial coordinates.
5. Barnes, S.L., 1964. A technique for maximizing details in numerical weather map analysis. Journal of Applied Meteorology and Climatology, 3(4), pp.396-409.

IWM-7 (22-26 March 2022)
IMD, MoES, NEW DELHI, INDIA